



Eidgenössische Technische Hochschule Zürich  
Swiss Federal Institute of Technology Zurich



# SWISS FEDERAL INSTITUTE OF TECHNOLOGY ZURICH

SEMESTER PROJECT SPRING 2017

DEPARTMENT OF INFORMATION TECHNOLOGY AND  
ELECTRICAL ENGINEERING

---

## Randomized Projections for Improved Sensing and Imaging in Positron Emission Tomography

---

*Author:*

Vesna RESENDE BARROS

*Under the supervision of:*

Matthieu SIMEONI, IBM

RESEARCH ZURICH

Dr. Paul HURLEY, IBM

RESEARCH ZURICH

Prof. Helmut BÖLCSKEI,

D-ITET, ETH

July 14, 2017

## Abstract

Positron Emission Tomography (PET) aims at recovering the metabolic activity of an organ of interest. Established algorithms implemented in contemporary PET scans are based on an approximation of the inverse Radon transform, resulting in a suboptimal estimate. In this context, the Bluebild algorithm [1,2] is proposed to recover the metabolic activity through a mathematical model that reformulates the inversion problem in a continuous framework. The procedure involves computing the inverse of a very large and dense Gram matrix, increasing significantly the computational cost and numerical instability of the algorithm.

In this project, we investigate the use of Gaussian random projections as means of reducing the high dimensionality of the PET scan data, and consequently of the Gram matrix. We show that the conditioning of the Gram matrix is improved in expectation, making the recovery more stable and resilient to noise. Simulations are used to assess the accuracy of the estimate as well as the conditioning of the Gram matrix. Finally, we show that the results with the Bluebild framework are more accurate than the state-of-the-art algorithms.

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
<b>2</b>	<b>The Physics of PET</b>	<b>4</b>
<b>3</b>	<b>Mathematical data model</b>	<b>5</b>
3.1	Links with the Radon Transform . . . . .	6
3.2	Computer Simulation of PET Data . . . . .	7
<b>4</b>	<b>Imaging the Data</b>	<b>8</b>
4.1	The filtered backprojection algorithm . . . . .	9
4.2	The Bluebild framework . . . . .	10
4.2.1	The Sampling Operator . . . . .	10
4.2.2	Interpolation . . . . .	11
4.2.3	Eigenvalue Decomposition of the Projection Operator .	12
4.2.4	The Gram matrix and the issues faced in the Bluebild algorithm . . . . .	13
<b>5</b>	<b>Randomized Projections and Data Compressing</b>	<b>15</b>
<b>6</b>	<b>Conclusion</b>	<b>23</b>
	<b>Appendix A Fundamentals of Projection Operators</b>	<b>26</b>
	<b>Appendix B Orthogonal and Biorthogonal bases</b>	<b>28</b>

# 1 Introduction

Positron emission tomography (PET) is a nuclear medicine functional imaging technique that is used to observe metabolic activity of an organ. A PET scanner is composed of an array of thousands of *scintillation detectors* arranged in a circular pattern around the patient's body. These sensors are used to record the  $\gamma$ -rays emitted when certain radioactive substance is injected into the patient. Estimating the metabolic activity of an organ can be seen as an *inverse problem* since we do not have direct access to the internal distribution, but only to the  $\gamma$ -rays recordings of the PET scan.

Originally introduced in the context of radio astronomy, the *Bluebird framework* [1, 2] developed by our team at IBM has also been successfully implemented in sound localization [3] and medical imaging, such as ultrasound [4] and PET [2]. In this project, we will focus on PET. The Bluebird framework conveniently models the acquisition system in terms of a sampling operator that maps continuous functions to discrete sets of measurements. A continuous least-squares estimate of the metabolic activity is then obtained by constructing an interpolation operator, generalized pseudoinverse of the sampling operator. Recovering the final estimate involves computing and inverting a Gram matrix, which has two drawbacks. First, the Gram matrix is very large in practice (on the order of  $10^6$  or more, depending on the number of scintillation detectors). Second, the Gram is very dense and structured, leading to ill-conditioning when inverting.

In this project, we propose the use of Gaussian random projections to reduce the dimensionality of the PET data, consequently fastening the least-squares recovery of the unknown function. We also show that these random compressions, in expectation, improve the conditioning of the Gram matrix and hence improve the numerical stability of the algorithm. Heuristics are used to determine the compression rate. Finally, we compare our imaging scheme to the state-of-the-art filtered backprojection algorithm.

In simulating the final PET images, the brain will be used as the organ of interest.

## 2 The Physics of PET

Before developing an imaging algorithm, we need to understand the theoretical foundations of PET imaging technique. This section will briefly introduce the PET principles of operation, and will permit us to further construct a data model.

PET is both a medical and research tool. It is used heavily in clinical oncology (medical imaging of tumors and the search for metastases) [5] and for clinical diagnosis of certain diseases such as Alzheimer's disease [6], Parkinson's disease [7], and coronary artery disease affecting heart muscle metabolism and flow [8].

The main goal of the PET is to access the metabolic activity of an organ of interest. The imaging process begins with the injection of a metabolically active tracer that carries with it a positron-emitting isotope. This radioactive tracer depends on the organ that is being analysed - for the brain, for example, these tracers are mixed with glucose (sugar), since it is the brain's main source of energy. Once injected in the patient, the tracer accumulates in areas of the body for which the molecule has an affinity, that is, in regions of highest metabolic activity [8–10]. Over a few minutes, the isotope decays by emitting a positron, which combines with an electron naturally present in the body's medium. This process, known as *annihilation*, generates two  $\gamma$ -rays that emerge from the body in opposite directions (see Figure 3.2). The radiation is then detected by an array of scintillation detectors placed around the patient and is recorded by the PET scan [8].

The resulting PET scan is an image showing the activity of the organ after radioactive tracers have been absorbed into the bloodstream. In the case of the brain, active areas consume glucose at a higher rate than inactive areas. When highlighted under a PET scanner, it allows doctors to see how the brain is working and helps them detect any abnormalities.

In practice, we cannot directly count the positron emissions, but only record the gamma radiation resulting from the annihilation of the positron with an electron. However, since the number of positron emissions is proportional to the metabolic activity, it is possible to indirectly access the unknown metabolic activity from the recorded data [8,9]. Thus, we say that tomographic imaging is an **inverse problem** in which the distribution of the positron-emitting tracer is inferred from the PET scan data after many annihilation events.

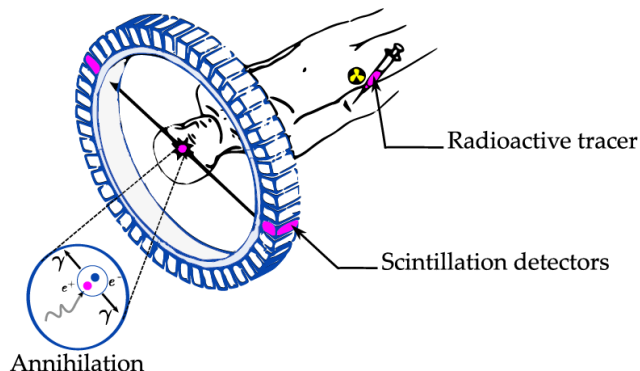


Figure 1: The physics of PET. A radioactive tracer is injected in the patient. Once it decays, the positron quickly annihilates with an electron, giving rise to two  $\gamma$ -rays emitted in opposite directions and detected by a pair of scintillation detectors that measures the function in different locations. Figure extracted from [11] with authorization from the authors.

### 3 Mathematical data model

The PET data model is intrinsically related to its principles of operation. When two  $\gamma$ -rays are recorded simultaneously by a pair of detectors, the annihilation event that gave rise to them must have occurred somewhere along the line connecting the detectors. In fact, we do not know exactly where in the line the positron was emitted, but if we are able to get a collection of measurements of the unknown function along all possible chords, then we could have an insight of how the distribution of positron emissions look like.

In an ideal case where there is infinite number of sensors, we could think about different lines sampling the internal metabolic activity, each one in a different spatial direction. In reality, however, there is a finite number of sensors (a typical PET facility contains around 3000 scintillation detectors). As a consequence, instead of lines, we observe *detector tubes* (see Figure 2) sampling the unknown distribution. Even though we cannot identify the exact location of the positron emissions, the best we can do is record the number of  $\gamma$ -rays coincidences in a detector tube  $d$ , denoted by  $n_d^*$ , and link both distributions to get a final estimate of the metabolic activity. Typically,  $n_d^*$  is modeled as a Poisson distributed random variable [11]:

$$n_d^* \sim \mathcal{P}(\lambda_d^*),$$

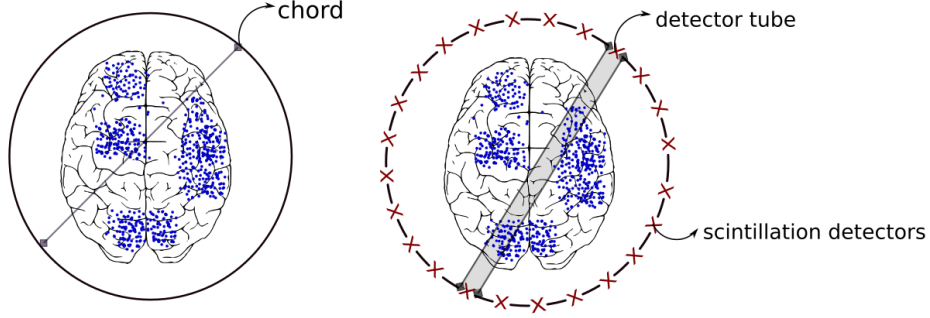


Figure 2: With infinite resolution, chords would be used to sample the metabolic activity of the brain (left). In reality, PET acquisition systems contain a finite number of sensors, meaning that instead of lines, detector tubes are defined to connect a pair of scintillation detectors in the ring (right).

where  $\lambda_d^*$  is the expected number of  $\gamma$ -rays in a detector tube  $d$ ,  $\lambda_d^* = \mathbb{E}[n_d^*]$ .

Finally, the link between the metabolic activity and the recorded PET scan data can be written as:

$$\lambda_d^* = \int_{\mathbb{R}^2} \lambda(x) \chi_d(p_d - \langle \mathbf{x}, \xi_{\phi_d} \rangle) d\mathbf{x}, \quad (1)$$

where  $(\phi_d, p_d) \in [0, \pi[ \times \mathbb{R}$  are the polar coordinates of the chord formed by the paired detectors (see Figure 3),  $\lambda(x)$  is the metabolic activity,  $\xi_{\phi_d} = (\cos(\phi), \sin(\phi))$  and  $\chi_d : \mathbb{R} \rightarrow \mathbb{R}$  is an indicator function, defining the  $d$ th detector tube with width  $\Delta \mathcal{P}_d > 0$ :

$$\chi_d(x) = \mathbb{1}\{|x| \leq \Delta \mathcal{P}_d\}, \quad d = 1, \dots, D.$$

Equation 1 can be seen as the inner product  $\langle \lambda, \chi_d \rangle$  that measures how much the unknown function resembles the basis functions  $\chi_d$ .

### 3.1 Links with the Radon Transform

The above data model is often simplified so as to interpret the data in terms of the **Radon Transform**. This transform takes a function  $f$  defined on the plane and maps it to a function  $\mathcal{R}f$  defined on the (two-dimensional) space of lines in the plane, whose value is the projection of the image intensity along a radial line oriented at a specific angle [10, 12].

In mathematical notation, let  $\mathbf{x} = (x, y) \in \mathbb{R}^2$  and consider a function  $f$  defined in some domain  $D \subset \mathbb{R}^2$ . The parametrization for a line  $L \subset \mathbb{R}^2$  is:

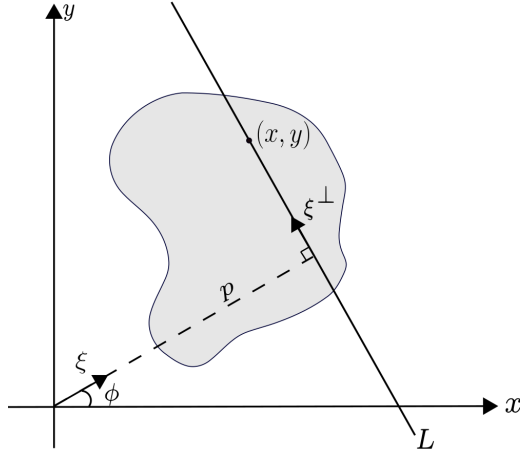


Figure 3: Visualization of the Radon transform in  $\mathbb{R}^2$  :  $\phi$  can take values in  $[0, \pi]$  and  $p$  is the distance from the origin to the line  $L$ .

$$L = \{\mathbf{x} \in \mathbb{R}^2 : \langle \mathbf{x}, \xi_{\phi_d} \rangle = p_d\}.$$

Then, using the Dirac delta function  $\delta(\cdot)$ , we can write the Radon transform as:

$$\mathcal{R}f(p, \xi_{\phi_d}) = \int_{\mathbb{R}^2} f(\mathbf{x}) \delta(\langle \mathbf{x}, \xi_{\phi_d} \rangle - p_d) d\mathbf{x}. \quad (2)$$

The Radon transform is the interpretation of equation 1 with the assumption of infinitely thin detectors, when the width of the detector tube  $\Delta\mathcal{P}_d \rightarrow 0$ . The intensity map of the entire data  $\mathcal{R}f(p, \xi_{\phi_d})$  is referred to as a **sinogram**.

### 3.2 Computer Simulation of PET Data

Since we did not have real data in hand, all the data in the subsequent experiments was simulated from the data model Eq. (1). To this end, a *phantom image* is used as a simplified metabolic activity. The phantom image (see Figure 4) is composed of ellipses of different sizes and gray levels, representing a simplified metabolic activity  $\lambda$  from which we wish to generate the data and that we wish to recover.

In the detector space, we chose a ring composed of  $N = 80$  scintillation detectors, which gives us  $D = N \times (N - 1)/2 = 3160$  detector tubes. In order



to simulate the Poisson distributed data  $n_d^*$ , Poisson noise was added to the original image. Moreover, the image is rescaled by a constant factor, so that the total number of positron emissions is equal to  $10^6$  (an approximation of the number of emissions during a PET scan session).

The sinogram is then obtained with equation 2. However, since it is not possible to acquire information coming from lines in all of the continuous possible directions (due to the finite number of sensors), one has to discretize in both  $p$  and  $\phi$  to get an approximation:

$$\Phi^H I = \begin{pmatrix} \sum_{i=1}^{Npix} I(r_i) \chi_1(r_i) \approx \langle I, \chi_1 \rangle \\ \vdots \\ \sum_{i=1}^{Npix} I(r_i) \chi_D(r_i) \approx \langle I, \chi_D \rangle \end{pmatrix},$$

where  $Npix$  is the number of pixels,  $I$  is the image,  $r_i$  is the corresponding pixel. The finite summation produces the sinogram on the right of the diagram in Figure 4, which is the output of the PET scan and that we will use to recover the metabolic activity  $\lambda$ .

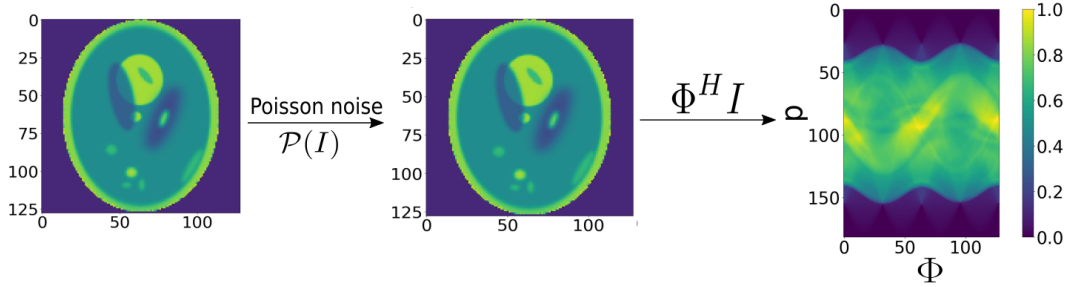


Figure 4: Simulation of an output of a PET scan. On the left, the original phantom image. In the middle, the same image generated with Poisson noise. On the right, the sinogram of the Poisson generated image. The PET simulation was implemented in Python with  $Npix = 256$ .

## 4 Imaging the Data

Now that we have a data model, we should understand how to get an estimate of the metabolic activity through the data. In this section, we will present the state-of-the-art filtered backprojection algorithm (FBP) as well as the Bluebird framework, to further compare the final estimate generated by the two algorithms.

## 4.1 The filtered backprojection algorithm

Contemporary PET scanners commonly make use of the inverse of the Radon transform for image reconstruction, known as the **filtered backprojection algorithm**. A detailed explanation on the algorithm can be found in [12, 13], but for now we will just state the discrete formula of the reconstructed function, known as the *discrete filtered backprojection*:

$$\lambda_{FB}(\mathbf{x}) = \frac{\Delta_\phi}{(2\pi)^2} \sum_{n=1}^{N_\phi} (\mathbf{h} \circledast \mathcal{R}f[\phi_n, \cdot]) \left[ \frac{\langle \mathbf{x}, \xi_{\phi_n} \rangle}{\Delta_p} \right], \quad (3)$$

where  $\mathcal{R}f \in \mathbb{R}^{N_\phi \times N_p}$  is a bi-dimensional sequence, interpolation of the Radon samples  $\mathcal{R}f(\phi_d, p_d)_d$  on a  $N_\phi \times N_p$  regular grid with steps  $\Delta_\phi$  and  $\Delta_p$ , and  $\mathbf{h}$  is a *Ramp filter* with DFT given by

$$\mathbf{H}[k] = \begin{cases} k, & 0 \leq k \leq \frac{N}{2} - 1, \\ (N - 1) - k, & \frac{N}{2} \leq k \leq N - 1. \end{cases}$$

By analogy with differentiation, the filter performs an operation similar to a derivative and is used to sharpen the recovered image. The discrete approximation in equation 3 is implemented in two phases, filtration and projection. Filtration consists of performing a DFT on a slice of the sinogram  $\mathcal{R}f(\phi_n, \cdot)$ , multiply it by the Ramp filter's Fourier transform and then use the inverse DFT to reconstruct the data. In the projection phase, the line integrals are projected back onto the plane at their respective angles. The discrete circular convolution approximation may lead some arbitrary values  $\langle \mathbf{x}, \chi_{\phi_n} \rangle$  to not fall exactly on one of the sinogram elements, so a further interpolation step - the simple *nearest-neighbours* interpolation - is required. Finally, the pixel intensities are summed to give the final reconstructed image (see figure 6).

The process of discretization can bring a few disadvantages. First, the ramp filter magnifies high frequency components of the image, which are usually dominated by noise. To counteract this, several other high-filters are commonly used (see [12, 14] for a full discussion, including

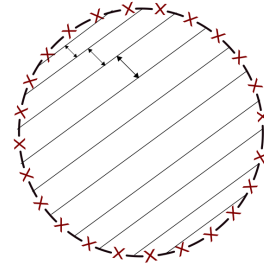


Figure 5: In reality, the distance between two consecutive sensors in the ring is non-uniform - the interval is shorter in the edges of the ring and longer in the center.

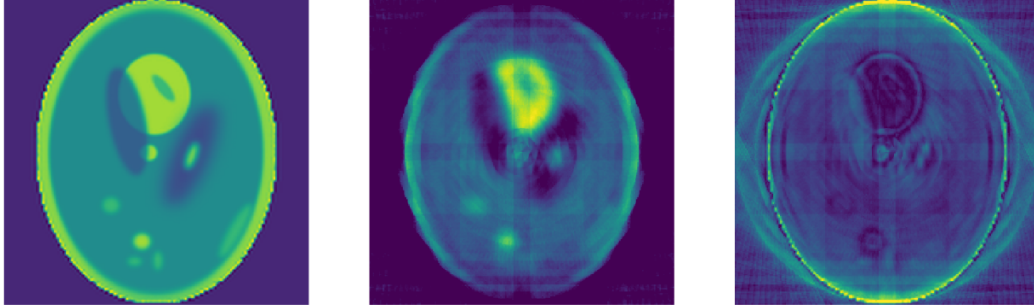


Figure 6: On the left, the original phantom image. In the middle, the filtered backprojection image, and the absolute error between them on the right.

derivations, on the use of filtration). Second, the spacing between two consecutive sensors in the PET ring is, in practice, non-uniform (see Figure 5 for a better visualization), which brings even more instability once it is interpolated on a uniform grid. For these reasons, the Radon inverse problem is **ill-posed**.

Statistical iterative algorithms are also used to reconstruct images in certain imaging techniques (see [15] for a review of the different approaches). Although iterative reconstructions have better performances, they are limited in resolution and computationally expensive. In general, the filtered backprojection is still the most widely used algorithm in PET scans nowadays.

## 4.2 The Bluebild framework

Now that we understand the reconstruction algorithm, how can we go further than the conventional medical imaging literature? In modeling the PET data, we do not want to add any other assumption to the model that is not naturally intrinsic to it. In other words, we aim at reformulating the problem in a continuous level, faithfully representing the physics phenomenon of positron emission tomography.

### 4.2.1 The Sampling Operator

Equation 1 showed the link between the desired metabolic activity and the provided PET scan data. We can reformulate this connection in terms of a **sampling operator** associated to the PET scanner:

$$n^* = \Phi^* \lambda, \quad (4)$$

where  $\Phi^* : \mathcal{H} \rightarrow \mathbb{R}^D$  is the sampling operator,  $n^* \in \mathbb{R}^D$  is the PET scan data,  $\lambda : \mathbb{R}^2 \rightarrow \mathbb{R}$  is the unknown metabolic activity and  $D$  is the number of detector tubes. The sampling operator works as an **analysis operator** acting on an infinite dimensional Hilbert space  $\mathcal{H} = \mathcal{L}^2(\mathbb{R}^2, \mathbb{R})$  to a finite dimension space  $\mathbb{R}^D$ , and sampling the unknown function  $\lambda$  over the family of functions  $\{\chi_d\}_d$ . Equation 4 can hence be interpreted as:

$$n^* = \Phi^* \lambda = \begin{bmatrix} \langle \lambda, \phi_1 \rangle \\ \vdots \\ \langle \lambda, \phi_D \rangle \end{bmatrix},$$

where  $\phi_d = \chi_d$ , as defined in section 3.2. Thus, the measurements  $n^*$  give us evidence about the components of  $\lambda$  present in  $\mathcal{R}(\Phi) = \text{span}(\phi_1, \dots, \phi_D)$ .

#### 4.2.2 Interpolation

The problem of estimating the metabolic activity from the measurements  $n^* \in \mathbb{R}^D$  is called *interpolation*. Mathematically speaking, we would like to map the finite measurements set back to an infinite dimensional estimate through a linear operator  $\tilde{\Phi} : \mathbb{R}^D \rightarrow \mathcal{H}$ , where  $\tilde{\Phi}$  is a **synthesis operator**, that we will conveniently call **interpolation operator** in all that follows. A requirement for choosing the interpolation operator is that it needs to be *consistent* with the sampling operator<sup>1</sup>:

$$\Phi^* \tilde{\Phi} = I$$

That is, sampling followed by interpolation needs to be the identity matrix. Since infinitely many interpolation operators are consistent, we need to further constrain it by  $\mathcal{R}\{\tilde{\Phi}\} = \mathcal{R}\{\Phi\}$ . Hence, we choose the interpolation operator  $\tilde{\Phi}$  to be the **unique generalized pseudoinverse** of  $\Phi^*$ :

$$\tilde{\Phi} = (\Phi^*)^\dagger = \Phi(\Phi^* \Phi)^{-1}, \quad (5)$$

where  $G_\Phi = \Phi^* \Phi \in \mathbb{R}^{D \times D}$  is the *Gram matrix*. When  $\tilde{\Phi}$  is chosen as in equation 5, the interpolation operator is said to be an *ideally matched* operator. In fact, we can show<sup>2</sup> that  $P = \tilde{\Phi} \Phi^*$  is an *orthogonal projection operator*, and gives us a good approximation of the metabolic activity in  $\mathcal{R}(\Phi)$ .

---

<sup>1</sup>See Appendix B for a detailed explanation on consistency, analysis and synthesis operators.

<sup>2</sup>Proven in Appendix B

The solution for the metabolic activity can thus be written as:

$$\lambda = \Phi(\Phi^*\Phi)^{-1}n^*, \quad (6)$$

which can also be reinterpreted in terms of an optimal estimate  $\hat{\lambda}$  in the least-squares sense:

$$\hat{\lambda} = \operatorname{argmin}_{\lambda \in \mathcal{R}(\Phi)} \|n^* - \Phi^*\lambda\|_2^2. \quad (7)$$

### 4.2.3 Eigenvalue Decomposition of the Projection Operator

In the section 4.2.4, we will see that when the Gram matrix is ill-conditioned, regularization is needed to make equation 6 well-defined. To understand the effect of regularization, we need to perform the eigenvalue decomposition of the projection operator  $P = \tilde{\Phi}\Phi^* = \Phi(\Phi^*\Phi)^{-1}\Phi^*$ . In other words, the solution  $\hat{\lambda}$  of equation 7 can also be written in terms of the eigendecomposition of  $P$ , that is,  $Pf = \mu f$ :

$$\underbrace{\Phi(\Phi^*\Phi)^{-1}\Phi^*}_{\alpha} f = \mu f \rightarrow \Phi\alpha = \mu f$$

where  $\alpha \in \mathbb{R}^N$ . Therefore,  $f \in \mathcal{R}(\Phi)$  and defining  $f = \Phi\alpha$  gives us

$$\underbrace{\Phi(\Phi^*\Phi)^{-1}\Phi^*\Phi}_{\mathcal{I}} \alpha = \Phi\alpha,$$

an eigenfunction with an eigenvalue  $\mu = 1$ . Choosing  $(\lambda_i, \alpha_i) \in \mathbb{R} \times \mathbb{R}^D$  as eigenpairs of  $G_\Phi$ , the eigenfunctions are orthogonal to each other:

$$\langle \Phi\alpha_i, \Phi\alpha_j \rangle \stackrel{(a)}{=} \langle \alpha_i, \Phi^*\Phi\alpha_j \rangle = \alpha_i^\top \Phi^*\Phi\alpha_j = 0,$$

where (a) comes from the definition of the adjoint operator. Moreover, the  $L_2$  norm of  $f$  is

$$\|\Phi\alpha_i\|_2 = \sqrt{(\Phi\alpha)^\top(\Phi\alpha)} = \sqrt{\alpha_i^\top \Phi^*\Phi\alpha_j} = \sqrt{\lambda_i}$$

Thus, normalizing  $f$  to unit norm, we can rewrite the least-squares estimate as:

$$\begin{aligned}
\Phi(\Phi^*\Phi)^{-1}\Phi^*\lambda &= \sum_{i=1}^D \mu_i f_i f_i^* \lambda \\
&= \sum_{i=1}^D \frac{(\Phi\alpha_i)}{\sqrt{\lambda_i}} \frac{(\Phi\alpha_i)^*}{\sqrt{\lambda_i}} \lambda \\
&= \sum_{i=1}^D \frac{\langle \Phi\alpha_i, \lambda \rangle}{\lambda_i} \Phi\alpha_i \\
&= \sum_{i=1}^D \frac{\langle \alpha_i, \Phi^*\lambda \rangle}{\lambda_i} \Phi\alpha_i \\
\hat{\lambda} &= \sum_{i=1}^D \frac{\alpha_i^\top n^*}{\lambda_i} \Phi\alpha_i
\end{aligned} \tag{8}$$

#### 4.2.4 The Gram matrix and the issues faced in the Bluebild algorithm

Each element of  $G_\Phi$  is given by

$$(G_\Phi)_{ij} = \langle \phi_i, \phi_j \rangle = \int_{\mathcal{B}} \chi_i(\mathbf{x}) \chi_j(\mathbf{x}) d\mathbf{x} \quad \forall i, j = 1, \dots, D.$$

Intuitively, the Gram matrix describes the coherence between the indicator functions of the tube. The inner product represents the area of the intersection of these bases (see Figure 7). Since they are very likely to intersect each other, the Gram is a very dense and non-sparse matrix, making its inversion numerically unstable. Moreover,  $G_\Phi$  is a large matrix ( $D \approx 50'000$ ), since its dimension is dependent on the number of detectors.

From equation 8, it is noticeable that for small variations in the measurements  $n^*$ , the final recovery can change significantly. Indeed, small eigenvalues  $\lambda_i$  amplify the measurement noise contained in  $n^*$ . To overcome ill-posedness in the inversion, a regularization method called **spectral truncation** is proposed:

$$\hat{\lambda}_{ST} = \sum_{i=1}^{\tau} \frac{\alpha_i^\top n^*}{\lambda_i} \Phi\alpha_i \tag{9}$$

with  $\tau \leq D$  some integer, *truncation parameter*.

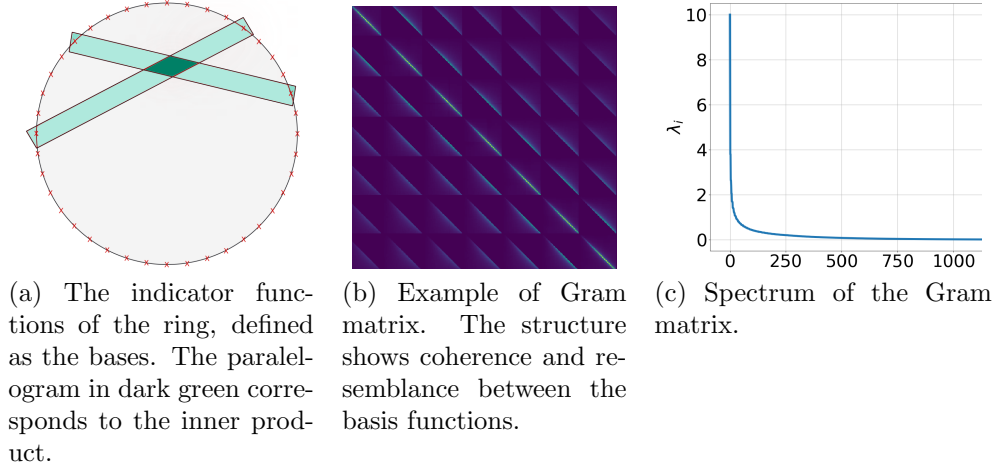


Figure 7: Properties of the Gram matrix.

Spectral truncation helps improving the conditioning number of the ill-posed system by truncating (getting rid of) the lowest eigenvalues responsible for the numerical instabilities. Figure 8 shows the regularized least-squares estimate in different situations: without the Gram correction, with Gram correction and no truncation, and with truncation in different levels. It's clear that the Gram correction is necessary for a better recovery of the image. In the case where there is no truncation, the small eigenvalues cause a negative effect in the final estimate due to the noisy measurements and the finite precision computing. When truncation is performed, the accuracy of the final image is dependent on the level of truncation - the more we truncate, the less accurate the recovery is since we are getting rid of eigenvalues responsible for the sharpness of the image. In general, the truncation parameter  $\tau$  is chosen using heuristics or visual inspection.

A comparison between the estimates of the filtered backprojection and the Bluebild algorithms can be found in Figure 9. Although the Bluebild approach gives us much more accurate images, the Gram matrix is still expensive to compute and invert. Thus, we investigate dimensionality reduction via random projections.

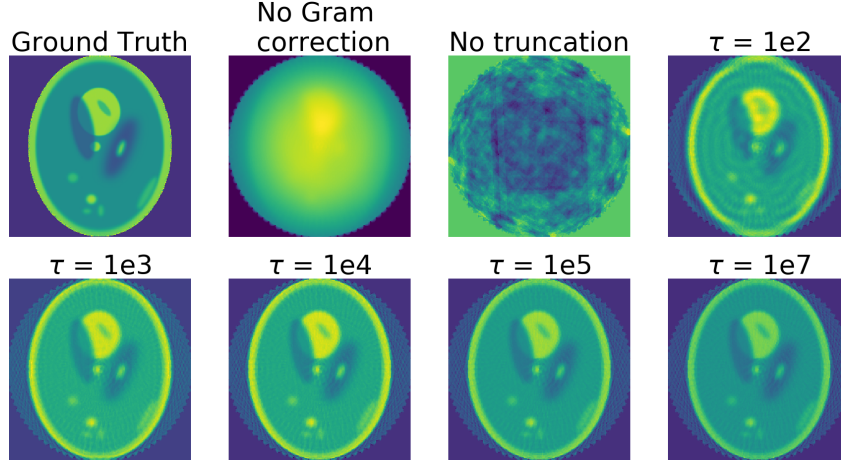


Figure 8: Recovered images using different truncation parameters. Without the Gram correction, the function is overestimated and no edges can be seen. With the Gram correction but without truncation, a blurred image is generated due to the presence of the small eigenvalues. The truncation parameter  $\tau$  is defined as the threshold  $\lambda_{max}/\tau$ . In other words, the highest the parameter  $\tau$ , less eigenvalues are eliminated and more accurate the images are.

## 5 Randomized Projections and Data Compressing

Although numerically stable, the regularization equation 9 is still very expensive to compute since it requires inverting the large matrix  $G_\Phi$  and performing its spectral decomposition. To fasten the algorithm, we propose to compress the data by means of random projections:

$$\tilde{\mathbf{n}} = W^\top \mathbf{n}^* = W^\top \Phi^* \lambda,$$

where  $W \in \mathbb{R}^{D \times L}$ ,  $L < D$  ( $L$  as a compression rate defined by the user) is a Gaussian Random matrix whose elements are given by:

$$(W)_{ij} \stackrel{i.i.d.}{\sim} \mathcal{N}(0, \sigma) \quad \forall i = 1, \dots, D, j = 1, \dots, L.$$

The introduction of this Gaussian matrix defines a new sampling operator  $\Psi^* : \mathcal{H} \rightarrow \mathbb{R}^L$

$$\Psi^* = W^\top \Phi^*$$



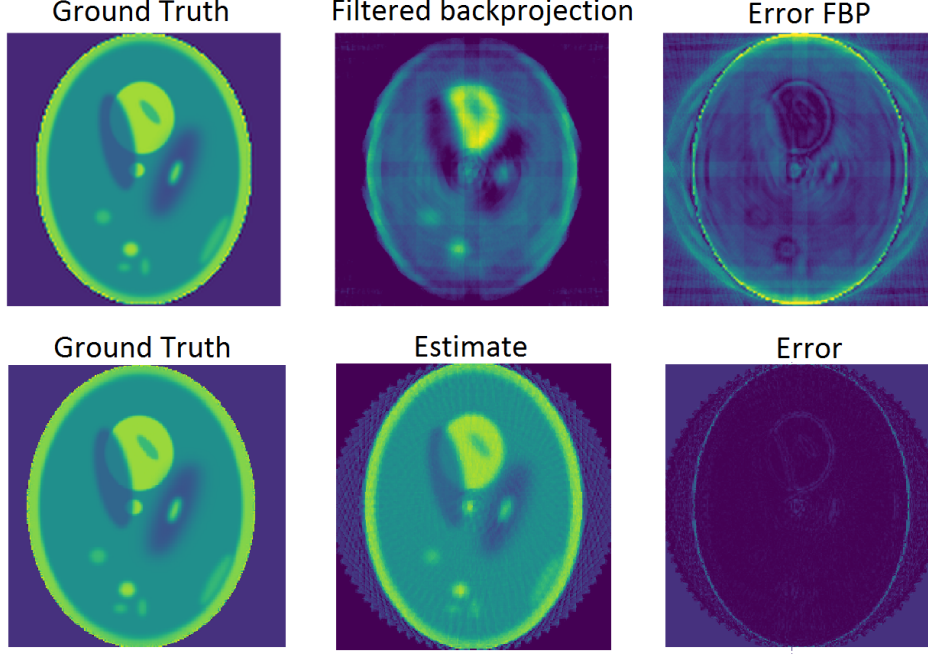


Figure 9: Comparison between the two algorithms showing the ground truth, the estimate and the error between them.

Moreover, each element  $W_{ij}$  is used to compute the new basis functions spanning  $\mathcal{R}(\Psi)$ :

$$\psi_j = \sum_{i=1}^D W_{ij} \phi_i, \quad \forall j = 1, \dots, L,$$

which gives us  $L$  new bases  $\{\psi_1, \dots, \psi_L\}$ . The compressed Gram matrix with random projections can be written as:

$$G_\Psi = \Psi^* \Psi = (W^\top \Phi^*) (W^\top \Phi^*)^* = W^\top \Phi^* \Phi W = W^\top G_\Phi W$$

where  $G_\Phi$  is the original Gram matrix  $\in \mathbb{R}^{D \times D}$ .

Examples of Gram matrices with and without random projections are shown in Figure 10. It's visible that after compression the Gram matrix looks very similar to a diagonal matrix, with not as much coherence between the basis due to the randomization. In fact, it can be proven that, on expectation, the Gram matrix  $G_\Psi$  is equal to the identity matrix if we choose the variance of the random Gaussian matrix  $W$  to be  $\sigma = 1/\text{trace}(G_\Phi)$ :

**Theorem 5.1.** (*Expected Gram matrix*) Assume  $G_\Phi \in \mathbb{R}^{D \times D}$  to be a symmetric, positive-definite matrix, and  $W \in \mathbb{R}^{D \times L}$ ,  $L < D$  a random Gaussian matrix with entries independent and identically distributed according to  $\mathcal{N}(0, \sigma)$ , where  $\sigma = 1/\text{trace}(G_\Psi)$ . Then, we have

$$\mathbb{E}[G_\Psi] := \mathbb{E}[\Psi^* \Psi] = \mathbb{E}[W^\top G_\Phi W] = I_L.$$

Proof (courtesy of [2]): Since  $G_\Phi$  is symmetric, positive-definite, it admits a Cholesky factorization:

$$G_\Psi = \Psi^* \Psi = W^\top G_\Phi W = W^\top C C^\top W = (C^\top W)^\top (C^\top W)$$

Denoting  $\Upsilon = C^\top W \in \mathbb{R}^{D \times L}$  and considering the vectorized versions of  $W$  and  $\Upsilon$  as:

$$\mathbf{w} := \text{vec}(W) \in \mathbb{R}^{DL}, \quad \mathbf{v} := \text{vec}(\Upsilon) \in \mathbb{R}^{DL}$$

By construction,  $\mathbf{w}$  is a random Gaussian vector with distribution:

$$\mathbf{w} \sim \mathcal{N}_{DL}(0, \sigma I_{DL})$$

Since  $\Upsilon = C^\top W I_L$ , we can write <sup>3</sup>

$$\mathbf{v} = (I_L \otimes C^\top) \mathbf{w},$$

where  $(I_L \otimes C^\top)$  is a  $DL \times DL$  real matrix. As a linear transform of a Gaussian random vector,  $\mathbf{v}$  is also a Gaussian random vector, with mean  $\mathbb{E}[\mathbf{v}] = (I_L \otimes C^\top) \mathbb{E}[\mathbf{w}] = 0$ , and covariance matrix given by

$$\begin{aligned} \text{Var}(\mathbf{v}) &= \mathbb{E}[\mathbf{v} \mathbf{v}^\top] \\ &= \mathbb{E}[(I_L \otimes C^\top) \mathbf{w} \mathbf{w}^\top (I_L \otimes C^\top)^\top] \\ &= (I_L \otimes C^\top) \mathbb{E}[\mathbf{w} \mathbf{w}^\top] (I_L \otimes C^\top)^\top \\ &= \sigma (I_L \otimes C^\top) I_{DL} (I_L \otimes C) \\ &= \sigma (I_L \otimes C^\top C) \\ &= \sigma (I_L \otimes G_\Phi). \end{aligned}$$

Furthermore,  $\sigma(I_L \otimes G_\Phi) \in \mathbb{R}^{DL \times DL}$  is a block-diagonal matrix of the form

---

<sup>3</sup>Recall that the vectorization is frequently used together with the Kronecker product to express matrix multiplication as a linear transformation on matrices. In particular,  $\text{vec}(ABC) = (C^\top \otimes A) \text{vec}(B)$

$$\sigma(I_L \otimes G_\Phi) = \sigma \begin{bmatrix} G_\Phi & 0 & \dots & 0 \\ 0 & G_\Phi & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & G_\Phi \end{bmatrix}$$

From the above equation it's possible to see that

$$\mathbb{E}[\Upsilon_{ij}^2] = \sigma(G_\Phi)_{ii}, \quad \forall i = 1, \dots, D, j = 1, \dots, L,$$

and

$$\mathbb{E}[\Upsilon_{ij}\Upsilon_{kl}] = \sigma(G_\Phi)_{ik}\delta_{jl}, \quad \forall i, k = 1, \dots, D, j, l = 1, \dots, L.$$

Now we can compute  $\varepsilon := \mathbb{E}[G_\Psi] = \mathbb{E}[\Upsilon^\top \Upsilon]$ . For the diagonal entries of  $\varepsilon$ , we have

$$\varepsilon_{ii} = \sum_{d=1}^D \mathbb{E}[\Upsilon_{di}^2] = \sigma \sum_{d=1}^D (G_\Psi)_{dd} = \sigma \text{trace}(G_\Psi) = 1$$

For the non-diagonal elements, we have

$$\varepsilon_{ij} = \sum_{d=1}^D \mathbb{E}[\Upsilon_{di}\Upsilon_{dj}] = 0, \quad \forall i \neq j.$$

Hence, we finally get

$$\varepsilon = \mathbb{E}[G_\Psi] = I_L,$$

wich achieves the proof.

In fact, we can check how the conditioning of the Gram improves with the random projections by analyzing Figure 13, where we observe the conditioning number approaching 1 for smaller dimensions. In practice, however, we may not have  $G_\Psi \simeq I_L$  since this is only true in expectation.

In Figure 11 we have computed our estimate using a single realization of the matrix  $W$  and different compression values. The compression rate ranges from smaller dimensions ( $L = 440$ ) until the original Gram matrix dimension ( $L = D = 3160$ ), where there is no compression at all.

A second experiment was investigated by repeating the random projections. We simulated 36 random matrices  $W_i$ , formed an image for each of them and summed them up to obtain the final estimate. A sequence of the generated images as the number of  $W$  matrices  $n$  increases is shown in Figure 12. During

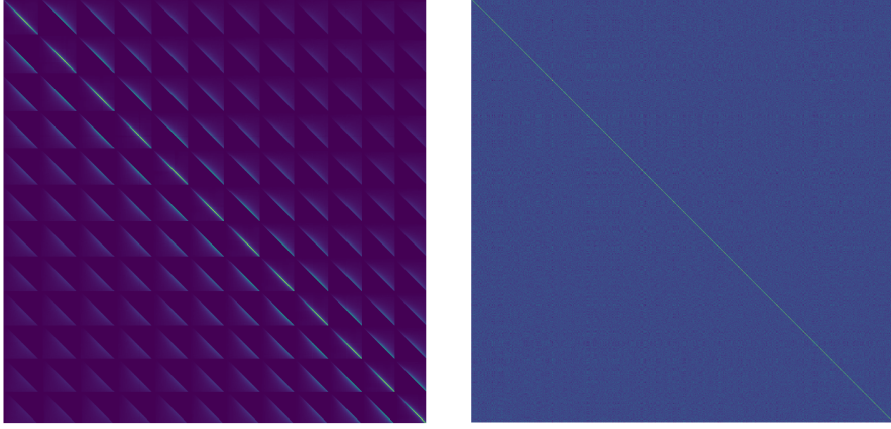


Figure 10: On the left, Gram matrix before random projections. We observe regular patterns due to the similarity between the basis. On the right, the compressed Gram matrix. The patterns disappear as a result of the random bases, leading to a less structured plot.

the first iterations, the improvement of the image is more perceptible - it gets sharper and sharper as we sum them up. After a certain point ( $n \approx 7$ ), the images don't seem to be changing too much. In fact, we can observe that the relative error between the ground truth and the estimate at each iteration rapidly decreases, reaching a *plateau* in the first iterations. For smaller dimensions ( $L = 100$  or  $200$ ), the error behaviour is somehow random, showing high peaks in the plot even for large values of  $n$ . In all simulations, a  $256 \times 256$  pixels image was used as the input image, and the number of sensors was set to  $N = 80$ .

In order to understand how the accuracy of the results varied as we compress the images, a third simulation was done by assessing the relative error of the generated figures (Figure 13a). In this case, each image was preconditioned with 20 different Gaussian matrices, and we plotted the mean relative error and the standard deviation for each compression level. For simulation purposes, we set the number of sensors to  $N = 40$ , resulting in an initial Gram matrix dimension of  $L = 780$ .

Practically speaking, one could make use of these results as a reference to check the best compression rate given a pre-established error. As we have seen, the correct choice of compression parameter  $L$  is very important to ensure a good reconstruction. In fact, the conditioning number of the compressed Gram changes drastically from small dimensions to big dimensions. This behaviour

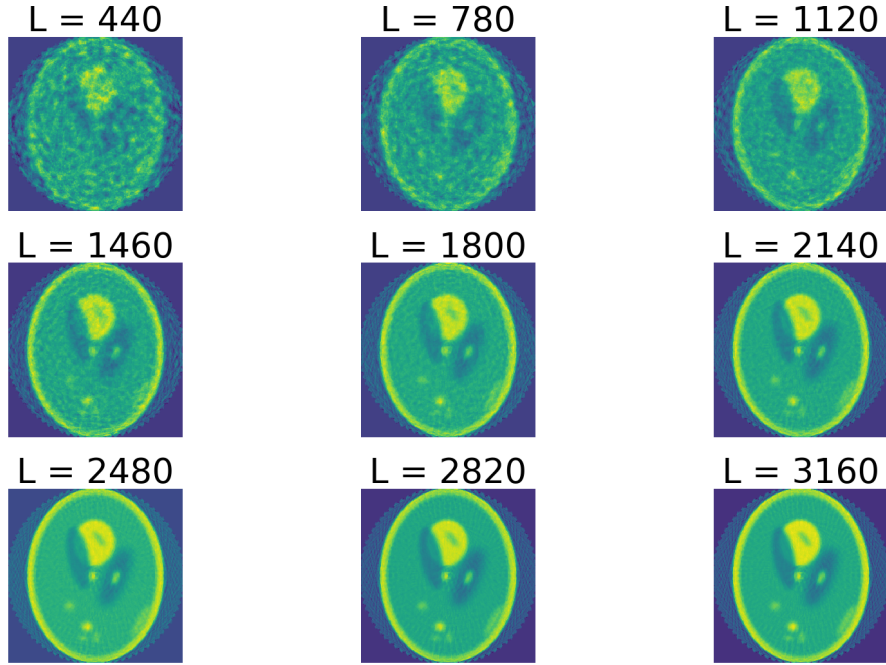
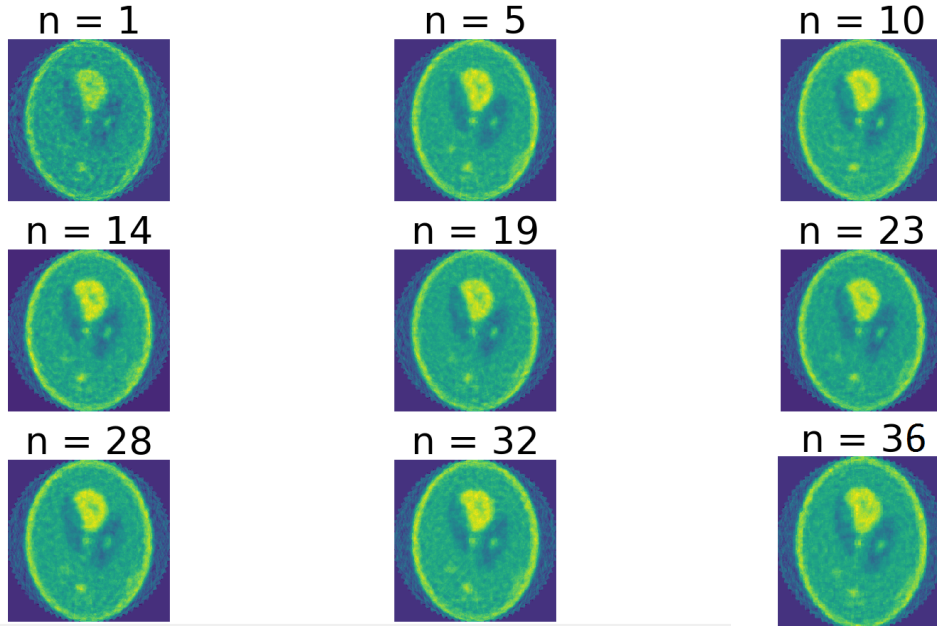
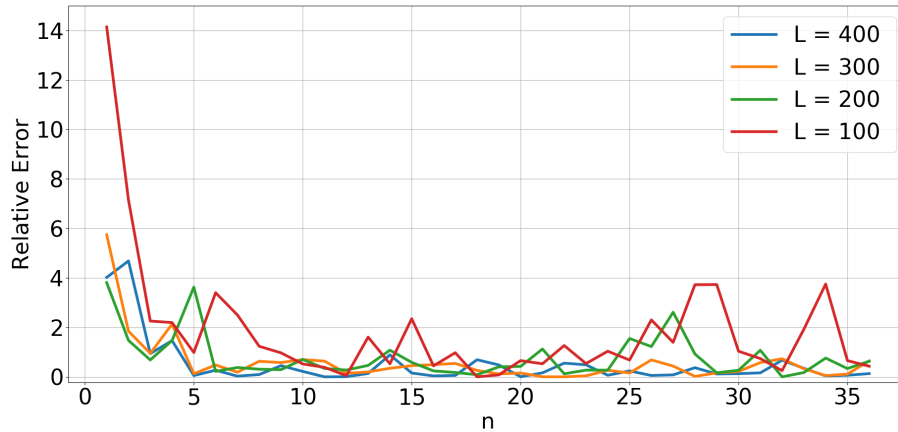


Figure 11: Estimates using compression rates ranging from small dimensions until the original dimension ( $L = D = 3160$ ). The more we compress, the less accurate the final image is.

is shown in Figure 13b, where the expected condition number of  $G_\Psi, \kappa(G_\Psi)$  is plotted as a function of the dimension. It's possible to observe that the conditioning of the compressed Gram is greatly improved for smaller dimensions, whereas for highest dimensions ( $L = 780$ ),  $\kappa(G_\Psi)$  reaches the original Gram matrix  $G_\Phi$ . In this case, spectral truncation (or any other regularization method) could help improving the conditioning of the system. All in all, we conclude that randomization only isn't sufficient to achieve a better conditioning of  $G_\Psi$  - we must also compress the data.

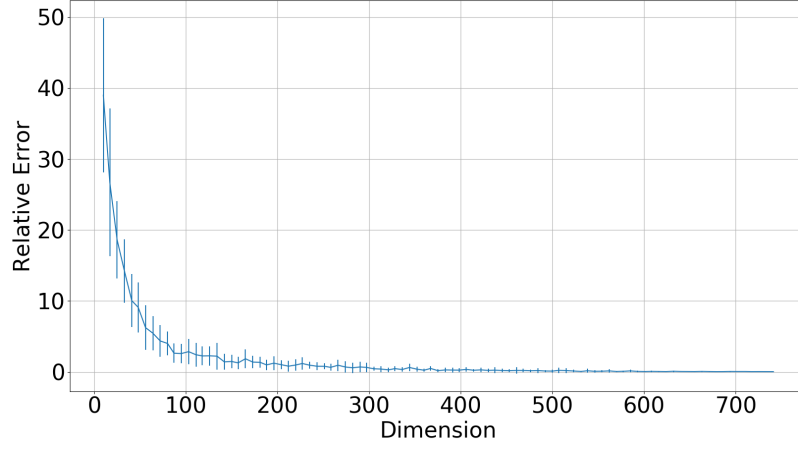


(a) The sequence of generated figures as  $n$ , the number of  $W$  matrices, increases. Each estimate is acquired by summing up the previous images.

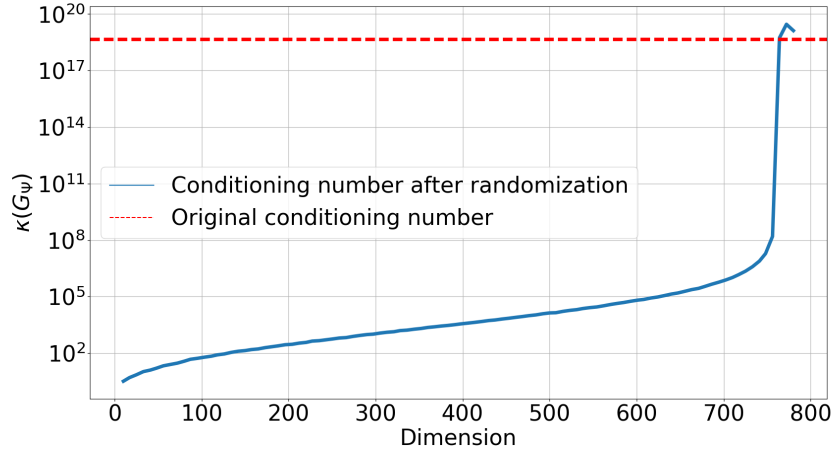


(b) The relative error of the images vs  $n$  for different compression levels  $L$ . In general, the error reaches a stable value after a few iterations. For smaller values of  $L$ , however, the error behaves more randomly.

Figure 12: Experiment with repeated random projections. The goal is to analyse the behaviour of the accuracy as we increase  $n$ , and investigate if adding more figures is actually bringing more accurate results.



(a) It's possible to observe the decay of the relative error as we compress less the images. Practically speaking, we can choose a certain compression rate  $L$  by visual inspection by setting a pre-established error.



(b) The behaviour of  $\kappa(G_\Psi)$ , the expected conditioning number of the Gram matrix, for different levels of compression. We observe the increase of  $\kappa(G_\Psi)$  as the compression level decreases, exceeding the original Gram matrix conditioning number.

Figure 13: Experiment with 20 repeated random projections with the purpose to analyze the accuracy and the conditioning number with respect to the compression level.

## 6 Conclusion

In this project, we investigated the use of Gaussian random projections to improve the computational efficiency of the Bluebild algorithm. Experimental and theoretical results confirm that these random projections indeed reduced the computational cost of the algorithm, and help improving the conditioning of the Gram.

Figure 13a showed that we could compress the data as high as 80% (from an initial dimension of 780 to 100) and still obtain a relative error of less than 10%. Such heuristics could be used in practice to obtain a good value of compression given a pre-established error level. Moreover, we also proved that the conditioning number is drastically improved by the random projections followed by compression.

The results obtained in the experiment in Figure 12 gave us an insight on how we can combine repeated projections. In fact, we can improve the error by averaging images, but we still need to understand how to combine it in an optimal way. For instance, the plot shows that after 7 or 8 iterations the error does not noticeably decrease anymore. However, this behaviour could change by investigating more advanced ways of combining the images. Future work may also include different randomization schemes that would make the algorithm even faster by producing sparse random matrices  $W$ , easier to manipulate.

## Acknowledgements

I would like to express my sincere gratitude to Paul Hurley for providing me an opportunity to do this semester project at IBM Research Zurich.

Also, I sincerely thank Matthieu for his guidance, patience and all the help during the period of my project work. I am really thankful for all the hours he dedicated to explain and discuss the mathematical concepts with me.

I also would like to thank Prof. Helmut Bölcskei for agreeing in supervise this project and making it possible to happen.

Finally, a big thank you for my team at IBM - Sepand, Lucien and Merve - for all the effort they put in our work and all the good times we spent together.



## References

- [1] M. M. J.-A. Simeoni, “Deconvolution of Gaussian Random Fields using the Graph Fourier Transform,” 2016, work done in collaboration with IBM Research, Zurich.
- [2] M. Simeoni, “Randomized beamforming for positron emission tomography,” 2016, unpublished manuscript.
- [3] S. Kashani, “Towards tractable high-resolution interferometric imaging with bluebild,” Master’s thesis, Swiss Federal Institute of Technology Lausanne, 2017.
- [4] L. Roquette, “A functional data analysis framework for ultrasound imaging,” Master’s thesis, Swiss Federal Institute of Technology Lausanne, 2017.
- [5] J. Bomanji, D. Costa, and P. Ell, “Clinical role of positron emission tomography in oncology,” *The lancet oncology*, vol. 2, no. 3, pp. 157–164, 2001.
- [6] R. Duara, C. Grady, J. Haxby, M. Sundaram, N. Cutler, L. Heston, A. Moore, N. Schlageter, S. Larson, and S. Rapoport, “Positron emission tomography in alzheimer’s disease,” *Neurology*, vol. 36, no. 7, pp. 879–879, 1986.
- [7] D. Calne, J. W. Langston, W. W. Martin, A. J. Stoessl, T. J. Ruth, M. J. Adam, B. D. Pate, and M. Schulzer, “Positron emission tomography after mptp: observations relating to the cause of parkinson’s disease,” *Nature*, vol. 317, no. 6034, pp. 246–248, 1985.
- [8] T. Budinger, F. Wehrli, S. Blumenfeld, F. Grunbaum, R. Henkelman, P. Lauterbur, W. Loeffler, F. Natterer, S. Nelson, L. Shepp *et al.*, “Mathematics and physics of emerging biomedical imaging,” *National Academy of Sciences, Washington DC*, 1996.
- [9] M. E. Phelps, J. Mazziotta, and H. R. Schelbert, “Positron emission tomography and autoradiography: principles and applications for the brain and heart,” 1985.
- [10] A. C. Kak and M. Slaney, *Principles of computerized tomographic imaging*. SIAM, 2001.

- [11] M. SIMEONI, “Statistical inference in positron emission tomography,” Ph.D. dissertation, Swiss Federal Institute of Technology Lausanne, 2014.
- [12] S. R. Deans, *The Radon transform and some of its applications*. Courier Corporation, 2007.
- [13] E. CANDÈS, “Lecture notes on applied fourier analysis and elements of modern signal processing,” Swiss Federal Institute of Technology Lausanne, 2016.
- [14] A. Hertle, “On the problem of well-posedness for the radon transform,” in *Mathematical Aspects of Computerized Tomography*. Springer, 1981, pp. 36–44.
- [15] A. Iriarte, R. Marabini, S. Matej, C. Sorzano, and R. Lewitt, “System models for pet statistical iterative reconstruction: A review,” *Computerized Medical Imaging and Graphics*, vol. 48, 2016.
- [16] M. Vetterli, J. Kovačević, and V. K. Goyal, *Foundations of signal processing*. Cambridge University Press, 2014.

## Appendix A Fundamentals of Projection Operators

The definitions and theorems summarized in this section are based on [16]. For a deeper and more detailed explanation, see the reference.

Let's start considering  $\mathcal{S}$  to be a closed subspace of a Hilbert space  $\mathcal{H}$  and  $x$  any vector in  $\mathcal{H}$ . The *least-squares approximation* is to find the vector in  $\mathcal{S}$  that is closest to  $x \in \mathcal{H}$

$$\hat{x} = \operatorname{argmin}_{s \in \mathcal{S}} \|x - s\|_2^2$$

Among all vectors  $s \in \mathcal{S}$ , the one that satisfies the equation above is uniquely determined by applying an *orthogonal projection operator* onto the vector  $x$ .

**Definition A.1** (Projection operator). A linear operator  $P$  is called a *projection operator* if:

- It is a bounded linear operator that is *idempotent*, that is, applying it twice is no different than applying it once.
- An *orthogonal projection* operator is a projection operator that is self-adjoint.
- An *oblique projection* operator is a projection operator that is not self-adjoint.

We also introduce the definition of the *adjoint operator*  $P^*$ .

**Definition A.2** (Adjoint and self-adjoint operator). The linear operator  $P^* : H_1 \rightarrow H_0$  is called the *adjoint* of the linear operator  $P : H_0 \rightarrow H_1$  when

$$\langle Px, y \rangle_{H_1} = \langle x, P^*y \rangle_{H_0}, \quad \text{for every } x \in H_0 \text{ and } y \in H_1$$

When  $P = P^*$ , the operator  $P$  is called *self-adjoint* or *Hermitian*.

The adjoint operator is a generalization of the conjugate transpose (or Hermitian transpose) in the infinite-dimensional linear algebra. Moreover, the adjoint  $P^*$  is always unique and the adjoint of  $P^*$  equals to the original operator  $(P^*)^* = P$

Returning to the least-squares problem, the unique solution  $\hat{x}$  can be written as a linear mapping  $\hat{x} = P_{\mathcal{S}}x$ , the orthogonal projection of  $x$  onto the subspace  $\mathcal{S}$ . Intuitively, we can prove it by noting that

$$P_{\mathcal{S}^\perp}x = x - P_{\mathcal{S}}x = x - \hat{x}$$

must be orthogonal to  $\mathcal{S}$ . Furthermore, for any vector  $s \in \mathcal{S}$ , we have

$$\|x - s\|^2 = \|(x - \hat{x}) + (\hat{x} - s)\|^2 = \|x - \hat{x}\|^2 + \|\hat{x} - s\|^2 \geq \|x - \hat{x}\|^2,$$

as a consequence of the Pythagorean theorem. Thus, the error is minimized when  $x = \hat{x}$ . Because  $\hat{x}$  is the orthogonal projection of  $x$  onto  $\mathcal{S}$ , the least-squares optimality of  $\hat{x}$  is known as the *Projection Theorem*. Alternatively, recognizing that the projection residual must be orthogonal to  $\mathcal{S}$ ,  $(x - \hat{x}) \perp \mathcal{S}$ , is known as the *Orthogonality Principle*:

**Theorem A.1.** *Orthogonality Principle*

*A bounded linear operator  $P$  in a Hilbert space  $\mathcal{H}$  satisfies*

$$\langle x - Px, Py \rangle = 0$$

*for all  $x, y \in \mathcal{H}$  if and only if  $P$  is an orthogonal projection operator.*

*Proof.* *The condition above is equivalent to:*

$$0 = \langle x - Px, Py \rangle \stackrel{(a)}{=} \langle P^*(x - Px), y \rangle = \langle P^*(I - P)x, y \rangle$$

*Where (a) comes from the definition of the adjoint. It implies that  $P^*(I - P) = P^* - P^*P = 0 \rightarrow P^* = P^*P$ . Assuming this holds, we have:*

$$P = (P^*)^* = (P^*P)^* = P^*P = P^*$$

*Thus,  $P$  is self-adjoint. Furthermore, we can show that  $P$  is also idempotent:*

$$P^2 = P^*P = P^* = P$$

*Thus,  $P$  is an orthogonal operator.*

## Appendix B Orthogonal and Biorthogonal bases

In a finite-dimensional vector space, a basis is a linearly independent set of vectors that is used to uniquely represent any vector in that vector space as a linear combination of the basis elements.

**Definition B.1** (Bases). The set  $\Phi = \{\varphi_{k \in \mathcal{K}}\}$ , where  $\mathcal{K}$  is finite or countably infinite, is a basis for  $\mathcal{H}$  when, for any  $x \in \mathcal{H}$ , there is a unique sequence  $\alpha$  such that

$$x = \sum_{k \in \mathcal{K}} \alpha_k \varphi_k, \quad (10)$$

where  $\alpha_k$  are the *expansion coefficients* of  $x$  with respect to the basis  $\Phi$ .

The subset  $\mathcal{S} = \text{span}\{\alpha_1 \varphi_1, \alpha_2 \varphi_2 \dots\}$  is referred to the subspace *spanned* by the vectors  $x$ . Alternatively, the span of  $\mathcal{S}$  is the set of all finite linear combinations of the elements of  $\mathcal{S}$ .

### Orthogonal bases

An orthogonal basis for a vector space is a basis whose vectors are mutually orthogonal. If the vectors of an orthogonal basis are normalized, the resulting basis is an *orthonormal basis*.

**Definition B.2.** (Orthonormal bases) The set of vectors  $\Phi = \{\varphi_{k \in \mathcal{K}}\}$ , where  $\mathcal{K}$  is finite or countably infinite, is called *orthonormal basis* or *standard basis* for a Hilbert space  $\mathcal{H}$  when

$$\langle \varphi_i, \varphi_k \rangle = \delta_{i-k},$$

for  $i, k \in \mathcal{S}$ , where  $\delta_{i-k}$  is the *Kronecker delta* sequence.

### Operators associated with bases

Given a family of basis  $\{\varphi_k\}$ , the expansion formula 10 can be viewed as linear mapping from a coefficient sequence  $\alpha$  to the vector  $x$ . In other words, expansion with a basis involves operations that can be expressed conveniently with matrices. We define two operators associated with the basis  $\{\varphi_k\}$ : the **analysis** and the **synthesis operators**.

**Definition B.3** (Basis Analysis Operator and Basis Synthesis Operator). Given a basis for a Hilbert space  $\mathcal{H}$ , the *analysis operator* associated with it is

$$\Phi^* : \mathcal{H} \rightarrow \mathbb{R}, \quad (\Phi^* x)_k = \langle x, \varphi_k \rangle, \quad k \in \mathcal{K} \quad (11)$$

Moreover, the *synthesis operator* associated with it is

$$\Phi : \mathbb{R} \rightarrow \mathcal{H}, \quad \Phi \alpha = \sum_{k \in \mathcal{K}} \alpha_k \varphi_k \quad (12)$$

### Basis expansion

Expansion coefficients with respect to an orthonormal basis can be obtained by using the same basis for signal analysis. Rewriting equation 10 as a function of the analysis operator gives us:

$$\alpha_k = \langle x, \varphi_k \rangle = \Phi^* x$$

Synthesis with these coefficients yields:

$$x = \sum_{k \in \mathcal{S}} \langle x, \varphi_k \rangle \varphi_k = \Phi \alpha = \Phi \Phi^* x \quad (13)$$

Therefore, for orthogonal basis, the recovery of any element in  $\mathcal{H}$  is done through equation 13.

Since  $x = \Phi \Phi^* x$  holds for all  $x \in \mathcal{H}$ ,  $\Phi \Phi^*$  must be the identity matrix  $I$  on  $\mathcal{H}$ :

$$\Phi \Phi^* = I \text{ on } \mathcal{H} \quad (14)$$

Furthermore,  $P = \Phi \Phi^*$  is an orthogonal projection operator in the case where  $\{\varphi_k\}$  is an orthogonal basis. To verify this, we show that  $P$  is self-adjoint and idempotent:

$$P^* = (\Phi \Phi^*)^* = (\Phi^*)^* \Phi^* = \Phi \Phi^* = P$$

Also,

$$P^2 = (\Phi \Phi^*)^2 = \Phi \Phi^* \Phi \Phi^* = \Phi (\Phi^* \Phi) \Phi^* \stackrel{(a)}{=} \Phi \Phi^* = P,$$

where (a) comes from the fact that

$$\Phi^* \Phi = I \quad (15)$$

Thus,  $P$  is an orthogonal projection operator.

Combined with with equation 14, equation 15 establishes that the analysis and synthesis operators associated with an orthonormal basis are unitary. To verify equation 15, we can do the following computation for any sequence  $\alpha$ :

$$\begin{aligned} \Phi^* \Phi \alpha &\stackrel{(a)}{=} \Phi^* \sum_{i \in \mathcal{K}} \alpha_i \phi_i \stackrel{(b)}{=} (\langle \sum_{i \in \mathcal{K}} \alpha_i \phi_i, \phi_k \rangle)_{k \in \mathcal{K}} \stackrel{(c)}{=} (\sum_{i \in \mathcal{K}} \alpha_i \langle \phi_i, \phi_k \rangle)_{k \in \mathcal{K}} \stackrel{(d)}{=} \\ &(\sum_{i \in \mathcal{K}} \alpha_i \delta_{i-k})_{k \in \mathcal{K}} \stackrel{(e)}{=} (\alpha_k)_{k \in \mathcal{K}} = \alpha \end{aligned}$$

where (a) follows from equation 12; (b) from equation 11; (c) from the linearity in the first argument of the inner product; (d) from the orthonormality of the set  $\{\phi_k\}$  and (e) from the definition of the Kronecker delta sequence.

## Nonorthogonal bases

Orthonormal bases have several advantages over nonorthonormal bases, including the simple expressions for expansion in equation 13. A basis does not have to be orthonormal to provide unique expansions. However, if the set of basis is not orthonormal, it is not possible for this single set to serve as both an analysis and synthesis role. This leads up to a new definition of *biorthogonal pair of bases*, or *dual bases*.

**Definition B.4.** (Biorthogonal pair of bases) The sets of vectors  $\Phi = \{\varphi_k\}_{k \in \mathcal{K}} \subset \mathcal{H}$  and  $\tilde{\Phi} = \{\tilde{\varphi}_k\}_{k \in \mathcal{K}} \subset \mathcal{H}$ , where  $\mathcal{K}$  is finite or countably infinite, are called *biorthogonal pair of bases* for a Hilbert space  $\mathcal{H}$  when

- each is a basis for  $\mathcal{H}$ ; and
- they are *biorthogonal*, meaning that

$$\langle \varphi_i, \tilde{\varphi}_k \rangle = \delta_{i-k} \quad \text{for every } i, k \in \mathcal{K}$$

The roles of the sets  $\Phi$  and  $\tilde{\Phi}$  can be reversed with no change. Here, we will generally maintain a convention of using the basis  $\Phi$  in analysis and the basis  $\tilde{\Phi}$  in synthesis. With each basis we associate synthesis and analysis operators defined through equations 12 and 11; a biorthogonal pair of bases thus yields four operators  $\Phi$ ,  $\Phi^*$ ,  $\tilde{\Phi}$ , and  $\tilde{\Phi}^*$ .

### Biothogonal basis expansion

With a biorthogonal pair of bases, expansion coefficients with respect to one basis are computed using the other basis. Analogously to equation 13, the expansion coefficients are computed with respect to the basis  $\Phi$ , and synthesis with these coefficients yields

$$x = \sum_{k \in \mathcal{S}} \langle x, \varphi_k \rangle \tilde{\varphi}_k = \tilde{\Phi} \alpha = \tilde{\Phi} \Phi^* x \quad (16)$$

Therefore, a biorthogonal pair of bases can together do the job of an orthonormal basis in terms of signal expansion. Since equation 16 holds for all  $x$  in  $\mathcal{H}$ ,

$$\tilde{\Phi} \Phi^* = I \text{ on } \mathcal{H} \quad (17)$$

More often, one wants all expansions to be with respect to one basis of the pair; the other basis of the pair serves as a helper in computing the expansion coefficients. The question now is: can we choose the synthesis operator  $\tilde{\Phi}$  to be a function of the basis  $\Phi$ ? If so, how do we determine  $\tilde{\Phi}$  from  $\Phi$ ?

### Consistency

A usual requirement for choosing a synthesis operator  $\tilde{\Phi}$  is that

$$\Phi^* \tilde{\Phi} = I \quad (18)$$

That is, synthesis followed by analysis needs to be equal to the identity. This is called **consistency** between the operators. Equation 17 shows that  $\Phi^*$  is a right inverse of  $\tilde{\Phi}$ . The consistency condition 18 makes  $\Phi^*$  a left inverse of  $\tilde{\Phi}$ , hence  $\tilde{\Phi}$  is the **unique generalized pseudoinverse** of  $\Phi$ , or the **generalised Moore-Penrose pseudoinverse**:

$$\tilde{\Phi} = (\Phi^*)^\dagger = \Phi(\Phi^* \Phi)^{-1} \quad (19)$$

When  $\tilde{\Phi}$  is chosen as equation 19,  $\Phi$  and  $\tilde{\Phi}$  span the same space  $\mathcal{H}$  and they're called *dual basis*. The expansion formula 16 for nonorthogonal basis can now be rewritten as



$$x = \Phi(\Phi^*\Phi)^{-1}\Phi^*x \quad (20)$$

For a finite set of basis  $\Phi$ , the matrix  $\Phi^*\Phi$  is called the **Gram matrix**, whose elements are given by

$$G_{ik} = \langle \varphi_k, \varphi_i \rangle, \quad \text{for every } i, k \in \mathcal{K},$$

where  $\mathcal{K}$  is finite. The elements of the matrix give us evidence about the similarity between the family of basis  $\{\varphi_k\}$ .

The computation of the Gram's inverse is a key virtue and a desired operator. It can, however, be very expensive to compute depending on the dimensions of the matrix.

In equation 16, we can show that  $P = \tilde{\Phi}\Phi^*$  is an orthogonal projection operator by verifying idempotency and self-adjointness:

$$P^2 = (\tilde{\Phi}\Phi^*)^2 = \tilde{\Phi}\Phi^*\tilde{\Phi}\Phi^* = \tilde{\Phi}(\Phi^*\tilde{\Phi})\Phi^* = \tilde{\Phi}\Phi^* = P$$

Thus,  $P$  is idempotent.

$$P^* = (\tilde{\Phi}\Phi^*)^* = (\Phi^*)^*\tilde{\Phi}^* = \Phi(\Phi(\Phi^*\Phi)^{-1})^* = \Phi((\Phi^*\Phi)^{-1})^*\Phi^*$$

$$\stackrel{a}{=} \Phi(\Phi^*\Phi)^{-1}\Phi^* = \tilde{\Phi}\Phi^* = P$$

where (a) follows from the fact that the Gram matrix  $(\Phi^*\Phi)$  is Hermitian symmetric.

In the case where  $\tilde{\Phi}$  is not given as equation 19, that is  $\tilde{\Phi} \neq \Phi(\Phi^*\Phi)^{-1}$ ,  $P$  is not an orthogonal projection operator and does not solve the least squares approximation. In this case,  $P$  is said to be an *oblique projection* (The concept goes beyond the scope of this project. For more information, see [16]).

**Example 1.** (Finite-dimensional orthogonal and nonorthogonal basis in  $\mathbb{R}^2$ )

The vectors  $e_1 = \begin{bmatrix} 1 & 0 \end{bmatrix}^\top$  and  $e_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}^\top$  form a standard basis for  $\mathbb{R}^2$ . To illustrate deviations from the orthonormal basis, we fix the vector  $x$  and vary  $e_1$  and  $e_2$  with an angle  $\theta = [0, \frac{\pi}{4})$ , as illustrated below. When  $\theta \neq 0$ , the bases become  $\tilde{e}_1$  and  $\tilde{e}_2$ .

We have seen that in the orthogonal case any vector  $x$  in  $\mathbb{R}^2$  can be perfectly recovered with  $\hat{x} = \Phi\Phi^*x$ , whereas in the nonorthogonal case the recovery is  $\hat{x} = \Phi(\Phi^*\Phi)^{-1}\Phi^*x$ . Moreover, the increase of  $\theta$  causes numerical conditioning

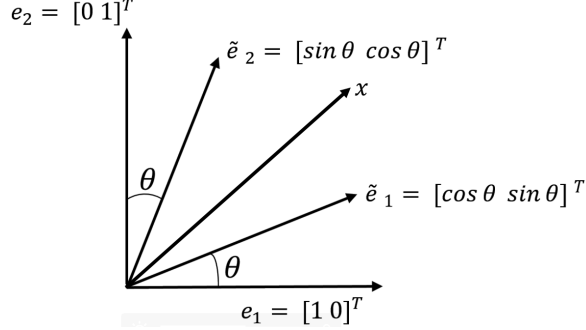


Figure 14: Illustration of example 1. The orthonormal bases  $e_1$  and  $e_2$  are varied with  $\theta$ , turning the system into nonorthogonal with bases  $\tilde{e}_1$  and  $\tilde{e}_2$ .

to be arbitrarily bad and the Gram matrix  $\Phi^* \Phi$  becomes ill-conditioned (high condition numbers)(see Figure 15a). We analyze the recovery accuracy of the vector  $x$  by computing the norm of the difference between the original vector  $x$  and the estimate  $\hat{x}$ . We calculate  $\hat{x}$  in two ways: using equations 13 (for orthogonal bases) and 20 (for nonorthogonal bases). In the latter, the vector  $x$  is perfectly recovered when  $\theta$  varies. In the first case, however, the loss of accuracy increases as  $\theta$  increases. Figure 15b illustrates this loss.

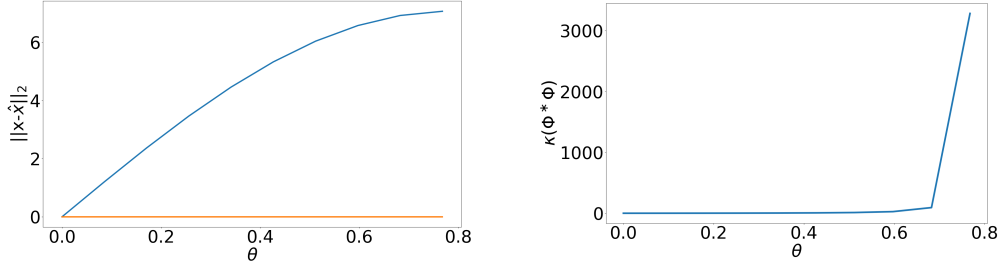
Numerical conditioning is ideal when  $\theta = 0$ , in which  $e_1$  and  $e_2$  is the orthonormal basis, while it is extremely poor for high values of  $\theta$ , resulting in very large expansion coefficients. As  $\theta$  approaches  $\pi/4$ , the basis vectors in  $\Phi$  are still noncolinear and form basis, but the numerical finite precision complicates the recovery.

**Example 2.** (Function recovery)

We define three different possible bases to use in a function recovery, as shown in Figure 16. The set of bases  $\{\varphi_k\}$  is now composed of all shifted versions of one single basis function, that can be a Gaussian, a window or a triangle function. The function  $f$  to be recovered can be any signal in  $\mathcal{L}^2(\mathcal{K})$ .

All estimates were calculated using Equation 20, since the overlap of the basis function creates a nonorthogonal basis. Three different functions  $f$  were tested to verify the performance of each basis in recovering  $f$ : sinusoid, piecewise-linear and sawtooth (triangular) function.

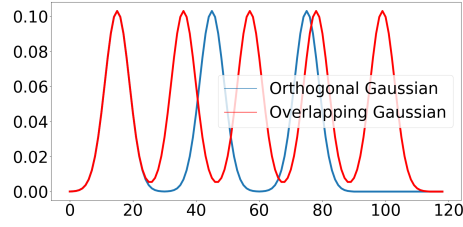
To verify the function recovery accuracy, the mean squared error (MSE) was calculated for each combination of function/basis. In general, Gaussian bases are better able to recover continuous and smooth functions, as can be seen in Fig. 17a. On the other hand, for piecewise functions with straight lines, Gaussian bases do not perform that well (Fig. 17b and 17c). The poor



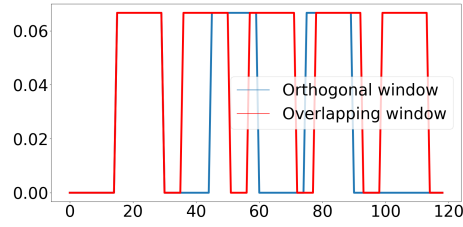
(a) Loss of accuracy in recovering  $x$ . The blue plot represents the increase of  $\|\hat{x} - x\|_2$  as  $\theta$  increases, when  $x$  is calculated with  $\hat{x} = \Phi\Phi^*x$ . The orange plot (constant zero valued) represents the norm calculated with  $\hat{x} = \Phi(\Phi^*\Phi)^{-1}\Phi^*x$ . (b) Gram matrix condition number. As  $\theta$  increases, the condition of the matrix  $\kappa(\Phi^*\Phi)$  also increases and the Gram becomes numerically ill-conditioned.

Figure 15: Recovery behaviour with orthogonal and nonorthogonal bases. On the left, the loss of accuracy is plotted for  $x$  recovered using the orthogonal basis equation (blue plot) and nonorthogonal basis equation (orange plot). On the right, the Gram matrix condition number is plotted as a function of  $\theta$ .

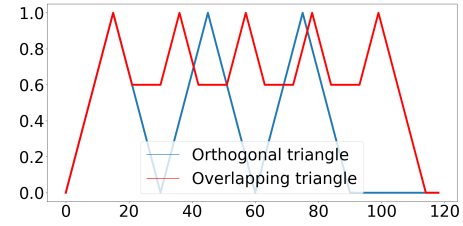
performance is more perceptible at the end of the line segments, where it is possible to visualize the Gibbs phenomenon - oscillations that increase in amplitude near the points of discontinuity. In contrast, the window basis is more appropriate to recover piecewise linear functions but does not reconstruct precisely the sine function (see Fig. 5d). The triangle basis presented relatively good results in recovering all functions. The results are presented in Table 1.



(a) Gaussian basis



(b) Window basis

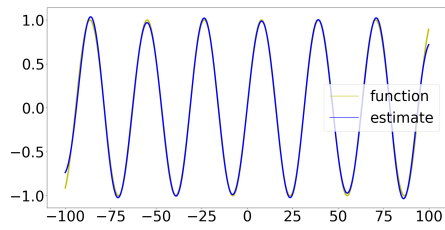


(c) Triangle basis

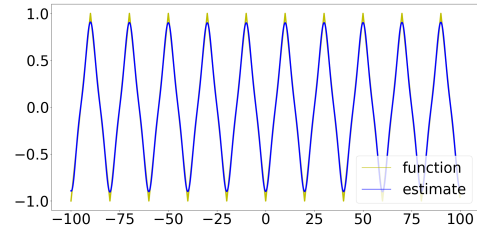
Figure 16: Examples of basis functions.

Table 1: Minimum Squared Error (MSE) for the recovered functions for different bases.

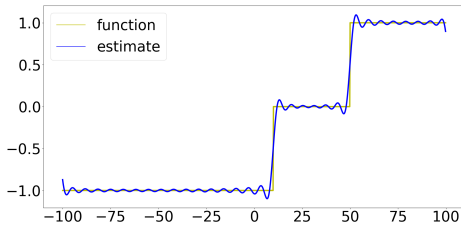
Function \ Basis	Gaussian	Window	Triangular
Piecewise-linear	$3.35 \times 10^{-3}$	$3.40 \times 10^{-2}$	$3.82 \times 10^{-3}$
Sine	$1.15 \times 10^{-3}$	$1.32 \times 10^{-2}$	$8.86 \times 10^{-3}$
Sawtooth	$8.15 \times 10^{-4}$	$1.33 \times 10^{-2}$	$5.36 \times 10^{-3}$



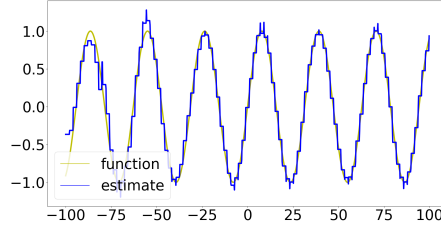
(a) Sine function recovered with Gaussian basis.



(b) Triangular function recovered with Gaussian basis.



(c) A piecewise function recovered with Gaussian basis.



(d) Sine function recovered with a window basis.

Figure 17: Approximations of different functions with Gaussian and window basis.